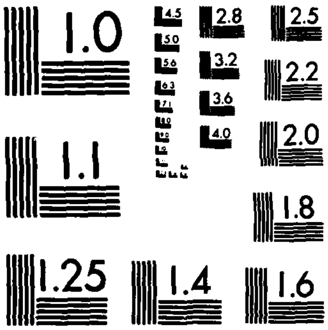
AD-A158 052 QUEUEING NETWORK SYSTEMS WITH UNBALANCED FLOWS AND THEIR APPLICATIONS TO..(U) ALFRED P SLOAN SCHOOL OF MANAGEMENT CAMBRIDGE MA CENTER FOR I.. Y R WANG ET AL. 1/1

UNCLASSIFIED AUG 84 CISR-M010-8408-14 N00039-81-C-0663 F/G 9/2 NL

END
FILMED
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(2)

AD-A158 052

Queueing Network Systems with Unbalanced Flows and Their Applications to Performance Evaluation of Highly Parallel Distributed Information Systems

Y. Richard Wang

Stuart E. Madnick

Technical Report #14

Revised August 1984

Center for Information Systems Research
Massachusetts Institute of Technology
Sloan School of Management
77 Massachusetts Avenue
Cambridge, Massachusetts, 02139

85 8 12 150

Contract Number N00039-81-C-0663
Internal Report Number M010-8408-14
Deliverable Number 2

Queueing Network Systems with Unbalanced Flows and Their Applications to Performance Evaluation of Highly Parallel Distributed Information Systems

Y. Richard Wang

Stuart E. Madnick

Technical Report #14

Revised August 1984

Center for Information Systems Research

Massachusetts Institute of Technology

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Technical Report #14	2. GOVT ACCESSION NO. AD-A158 052	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Queueing Network Systems with Unbalanced Flows and Their Applications to Performance Evaluation of Highly Parallel Distributed Information Systems		5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Yng-Yuh Richard Wang Stuart Madnick		8. CONTRACT OR GRANT NUMBER(s) N00039-81-C-0663
9. PERFORMING ORGANIZATION NAME AND ADDRESS Center for Information Systems Research Sloan School of Management, M.I.T. Cambridge, MA 02139		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Electronics Systems Command		12. REPORT DATE August 1984 13. NUMBER OF PAGES 36
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

simulation, queueing network, unbalanced flows, software engineering, architectural design, data storage hierarchy, intelligent memory, data base computer, distributed information systems

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

An information system analysis methodology using certain aspects of queueing theory to evaluate computer system speed performance early in the design process is presented. Specifically, a methodology is presented to compute performance measures for distributed systems with unbalanced flows (i.e., number of transactions entering that server) due to asynchronously spawned paralled tasks--an increasingly important phenomenon in

modern information systems which has a significant effect on performance that cannot be analyzed by classical queueing network models.

A decomposition method is applied to decompose the unbalanced flows. Formulae for open queueing networks with unbalanced flows due to asynchronously spawned tasks are developed. Furthermore, an algorithm based on Buzen's convolution algorithm is developed to test the necessary and sufficient condition for closed system stability as well as to compute performance measures. An average of less than four iterations is necessary for convergence with this algorithm.

A study of the INFOPLEX data storage hierarchy has been conducted using both this rapid solution algorithm and detailed simulations; highly consistent results were obtained. A cost effective software tool, using this methodology, has been developed to analyze an architectural design and to produce measures such as throughput, utilization, and response time so that potential performance problems can be identified.

ABSTRACT:

An information system analysis methodology using certain aspects of queueing theory to evaluate computer system speed performance early in the design process is presented. Specifically, a methodology is presented to compute performance measures for distributed information systems with unbalanced flows (i.e., number of transactions leaving a server is not the same as number of transactions entering that server) due to asynchronously spawned parallel tasks -- an increasingly important phenomenon in modern information systems which has a significant effect on performance that cannot be analyzed by classical queueing network models.

A decomposition method is applied to decompose the unbalanced flows. Formulae for open queueing networks with unbalanced flows due to asynchronously spawned tasks are developed. Furthermore, An algorithm based on Buzen's convolution algorithm is developed to test the necessary and sufficient condition for closed system stability as well as to compute performance measures. An average of less than four iterations is necessary for convergence with this algorithm.

A study of the INFOPLEX data storage hierarchy has been conducted using both this rapid solution algorithm and detailed simulations; highly consistent results were obtained. A cost effective software tool, using this methodology, has been developed to analyze an architectural design and to produce measures such as throughput, utilization, and response time so that potential performance problems can be identified.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 SIGNIFICANCE OF THE RESEARCH

This paper models Unbalanced flows due to Asynchronously spawned Parallel tasks (UAP) in a distributed system using generalized queueing network models as developed by Jackson (1963), Gordon and Newell (1967), Baskett et al (1975), Reiser and Kobayashi (1975), and Denning and Buzen (1978). The acronym UAP will be used throughout the paper to refer to unbalanced flows (i.e. number of transactions leaving a server is not the same as number of transactions entering that server) due to asynchronously spawned parallel tasks which are assumed to run independently of each other except for resource contention.

The significance of this research is as follows: (1) A decomposition method is applied to incorporate the workload due to UAP -- a primary effect on speed performance -- into queueing network models; (2) Formulae for open queueing networks with UAP are developed; (3) An algorithm based on Buzen's convolution algorithm is developed to test the necessary and sufficient condition for closed system stability; and (4) Formulae for closed queueing networks with UAP are developed based on the algorithm. As a result, performance measures are assessed more accurately. A cost effective software tool, using this methodology, has been developed (Wang and Madnick, 1984) to

analyze system architectures with UAP so that performance implications can be identified early in the design process.

The discussion in this paper uses as an example the INFOPLEX data storage hierarchy (INFOPLEX is a database computer research project at the MIT Center for Information Systems Research; the theory of hierarchical decomposition is applied in this research to structure hundreds of microprocessors together to realize a low cost data storage hierarchy with very large capacity and small access time. See Madnick, 1973, 1977, 1979, Lam and Madnick, 1979, and Wang and Madnick, 1981, 1984). However, the methodology employed in this research is generalizable to other distributed information systems (Trivedi and Sigmon, 1981, Geist and Trivedi, 1982, and Goyal and Agerwala, 1984).

1.2 PURPOSE AND BACKGROUND OF THE RESEARCH

The goal of system development is to produce systems that satisfy their specifications when completed while minimizing costs and time required. A key to minimizing development cost and time is to determine whether the system will meet its functional and performance requirements as early as possible in the development process. This will avoid wasted work toward an unsatisfactory implementation and the subsequent rework. To this end, a cost effective tool to evaluate system performance is essential (Gagliardi, 1982).

The use of analytic performance models, instead of simulation models, has become increasingly popular recently

(Graham, 1978) because the analytic approach is more cost effective than simulation. Figure 1-1 illustrates the difference in terms of CPU time and dollar cost between the analytic model and the simulation model that the authors have conducted for a 5 processor 4 level INFOPLEX design, called P5L4. The cost of running the analytic model was found to be from 200 to 2000 times less expensive (Wang and Madnick, 1984). An even more important practical difference was that a single simulation run, to test one design of the fairly simple P5L4 system, costs about $100 -- whereas, the comparable analytic calcuation costs around 5 cents. At $100 or more per design, it may be difficult for a researcher to justify the costs of exploring the hundreds or thousands of design and parameter alternatives. At 5 cents per design studied, using the analytic model, the researcher can freely explore many diverse alternatives and a wide range of parameters in search for an optimal design.

In order to obtain accurate results, all significant factors effecting performance should be captured in the analytic model. UAP has been found to have a primary effect on performance (Wang and Madnick, 1981, 1984). Unfortunately, networks with UAP did not have an analytically tractable solution because the input flow and the output flow are not balanced at the places where parallel tasks are spawned, a violation of the principle of job flow balance (Denning and Buzen, 1978: the principle of job flow balance says that the number of customers that flow into a service facility equals to the number of customers that flow out of the facility when the system is in the steady-state).

RUN	Simulation PERIOD	Simulation CPU-TIME	Simulation COST$	Analytic CPU-TIME	Analytic COST$
1	10 ms	434	97.33	12	0.05
2	3 ms	270	61.70	12	0.05
3	2 ms	349	78.22	12	0.05
4	2 ms	308	70.32	12	0.05
5	1 ms	205	47.77	12	0.05
6	1 ms	351	79.02	12	0.05
7	.5 ms	453	101.06	12	0.05
8	.3 ms	290	65.55	12	0.05
9	.05 ms	47	13.09	12	0.05
10	.05 ms	38	10.54	12	0.05

Simulation CPU-TIME is in CPU seconds on an IBM 370/168.
Analytic CPU-TIME is 12 CPU seconds per run on a PRIME/850.
"Cost$" is in dollars for the overall charge per run.
"ms" in the table means milli-seconds.
To attain steady-state, simulation periods of 10 ms, or more, are usually needed.

Figure 1-1: A Comparison of costs: Simulation vs. Analytic.

A simplified 1 processor 2 level INFOPLEX model, called P1L2, is given below to illustrate the UAP phenomenon. Consider the routing diagram (Figure 1-2) of the P1L2 model which processes read and write operations. Suppose 80% of the customers request the read operation (RP1) and 20% request the write operation (WP1); and the read operation has 100% locality, i.e. the requested data is always found at the level one device, D1. The read operation is serviced by the level one processor, P1, first, then the data is retrieved from D1 and returned to the reference source (SINKM). The write operation is acknowledged immediately by P1 to the reference source (SINKM);

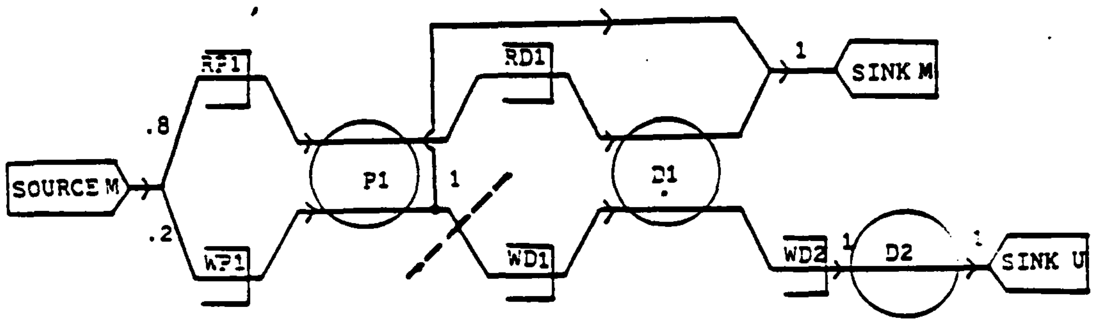


Figure 1.2 Routing Diagram for P1L2 Model

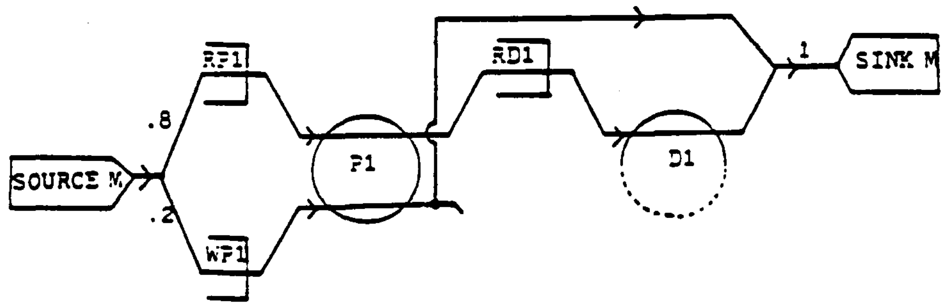


Figure 1.3 Main Chain

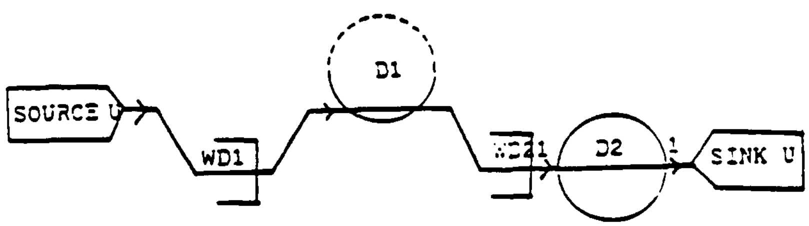


Figure 1.4 UAP Chain

in parallel, the data is updated at D1, stored-behind to the level 2 device, D2, then the asynchronously spawned task terminates (SINKU).

Note that class WP1 leaves facility P1 with "a routing probability one" to SINKM and "a routing probability one" to WD1 as indicated by the dash line; i.e. the out-flow is twice as much as the in-flow, violating the principle of flow balance.

1.3 LITERATURE REVIEW

Several studies have been attempted to generalize queueing network models to include parallel processing. Browne, Chandy, Horgarth, and Lee (1973) investigated the effect on throughput of multiprocessing in a multiprogramming environment using the central server model approach. Reiser and Chandy (1979) studied the impact of distributions and disciplines on multiple processor systems. Towsley, Chandy, and Browne (1978) developed approximate queueing models for internal parallel processing by individual programs in a multiprogrammed system based on the central model approach and "Norton theorem". Price (1975) analyzed models of multiple I/O buffering schemes. Maekawa (1976) and Peterson (1979) modeled a number of CPU:IO overlap cases. These studies, although valuable, do not fit systems which have a generalized topology and the UAP phenomenon.

Modeling the UAP phenomenon for generalized queueing network systems is a relatively new topic, first reported, to our knowledge, by Heidelberger and Trivedi in 1982. In that work, an

Proof: Define (Reiser and Kobayashi, 1975) the p.g.f. for

$P(n_1(M), n_1(U), \ldots, n_M(M), n_M(U))$ as

$$G(Z,\theta) = \Pi\ \Phi_i(\ \rho_i(U) * z_i(U) + \rho_i(M) * z_i(M) * \theta)$$

where z_i is the p.g.f. transformation variable for facility i; θ is a factor associated with the main chain to insure that main chain population is fixed to N; the product, Π, is taken from 1 up to M, and $\Phi_i(\zeta) = 1/(1-\zeta)$ for FCFS, PS, and LCFSPR. The p.g.f. is found as the coefficient of θ^N in a power series expansion of $G(Z,\theta)$ in θ, denote it $G^*(Z)$. It follows that

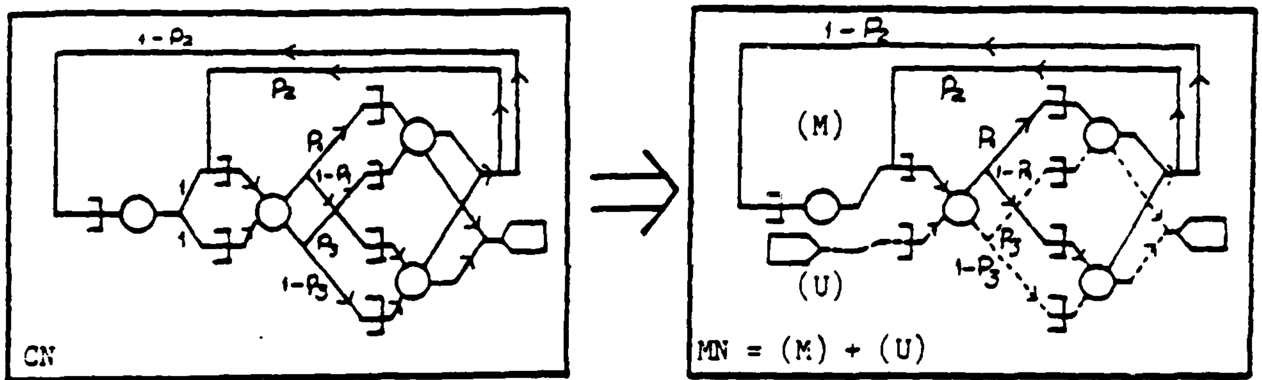
$$G^*(Z) = C * \partial_\theta(N) * \Pi\ \Phi_i(\rho_i(U) * z_i(U) + \rho_i(M) * z_i(M) * \theta)$$

To obtain the p.g.f. of the marginal distribution of the closed main chain, let $z_i(U)=1$. It follows that

$$G^*\ (z_i(U)=1)$$
$$= C * \partial_\theta(N) * \Pi\ \Phi_i\ (\rho_i(U) + \rho_i(M) * z_i(M) * \theta)$$
$$= C * \partial_\theta(N) * \Pi\ 1/(\ 1 - \rho_i(U) - \rho_i(M) * z_i(M) * \theta\ )$$
$$= C * (\Pi\ 1/(1 - \rho_i(U))) * \partial_\theta(N)$$
$$* \Pi\ 1/(1-(\ \rho_i(M) * z_i(M) * \theta\ /\ (\ 1 - \rho_i(U))))$$
$$= (1/G(N)) * (\ \Sigma\Pi\ (\rho_i(M) * z_i(M)\ /(\ 1 - \rho_i(U)))^{n_i(M)})$$

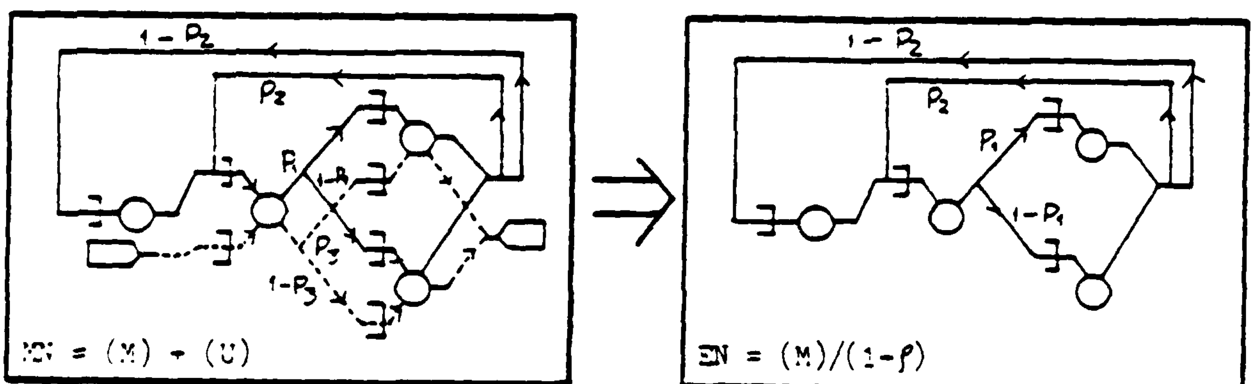
where the summation is taken over all possible states of $S(N,M) = \{\ (n_1(M), \ldots, n_M(M)) \mid n_1(M) + \ldots + n_M(M) = N$, and $n_i(M) \geq 0$ for all $i\ \}$. But this is exactly the p.g.f. for CPFSCQN (Lavenberg, 1983) with the traffic intensity inflated by $(1-\rho_i(U))^{-1}$ for facility i. Q.E.D.

From the marginal distribution above, it is not difficult to show (Heidelberger, 1982, Mayrhauser, 1983) that f is CMD, assuming that there exists at least a pair of $(D_i(M), D_i(U))$ such that $D_i(M)>0$ and $D_i(U)>0$. With the corollary and the CMD



Original Closed Network with UAP(CN)   Mixed Network without UAP(MN)

<u>Figure 3.1</u>   <u>Decomposition of CN to MN</u>



Mixed Network without UAP(MN)   Equivalent Network without UAP(EN)

<u>Figure 3.2</u>   <u>Transformaion of MN to EN for the Main Chain</u>

3.4 CLOSED QUEUEING NETWORKS WITH UAP

For closed queueing networks with UAP, a mixed network with the closed main chain and the open UAP chain, as illustrated in Figure 3-1, can be obtained following the discussion in section 3.1. Since $X_0(U) = X_0(M) * V(U)$ where $X_0(M)$ is evaluated through a nonlinear function of $X_0(U)$ (Reiser and Kobayashi, 1975). It follows that $X_0(U) = f(X_0(U)) * V(U)$ where f is a nonlinear function. To solve the nonlinear equation, a couple of issues have to be addressed first:

A) What are the properties of f?

B) what is the necessary and sufficient condition for the network to be stable?

A corollary based on Reiser and Kobayashi's theorem (1975) on PFMQN is shown below to settle issue A; and a lemma is proven to settle issue B which leads to an iterative procedure for the closed network. The IS discipline is excluded from this subsection. Its difference from other disciplines will be discussed at the end of the section.

A) **Corollary:** An equivalent closed network(EN) of the main chain for the mixed network(MN), as illustrated in Figure 3-2, can be obtained by inflating the main chain traffic intensities, i.e. by replacing $\rho_i(M)$ by $\rho_i(M) / (1-\rho_i(U))$ for i = 1, ..., M.

It can be shown (Sauer 1981, Lazowska 1984) that throughput, utilization, mean queue length, and response time are computed as shown in Table 3-1. Note that: 1). $U_i(M)$ would be the sum of the products of throughputs and mean service times of all the classes if there were multiple classes of customers at facility i for the main chain; the same mechanism applies to the UAP chain. 2). The denominator of $N_i(M)$ is U_i which quantifies the resource contention between the UAP chain and the main chain. 3). The "system response time" in a flow unbalanced network is defined as the time to complete the main chain, $R_o(M)$, since that is the only observable completion seen by the external world.

Facility i	FCFS,PS,LCFSPR discipline
$X_i(M)$	$X_o(M) * V_i(M)$
$X_i(U)$	$X_o(U) * V_i(U) / V(U)$
X_i	$X_i(M) + X_i(U)$
$U_i(M)$	$X_i(M) * S_i(M)$
$U_i(U)$	$X_i(U) * S_i(U)$
U_i	$U_i(M) + U_i(U)$
$N_i(M)$	$U_i(M) / (1-U_i)$
$N_i(U)$	$U_i(U) / (1-U_i)$
N_i	$N_i(M) + N_i(U)$
$R_i(M)$	$N_i(M) / X_i(M)$
$R_i(U)$	$N_i(U) / X_i(U)$
R_i	N_i / X_i
$R_o(M)$	$R_1(M) + \ldots + R_C(M)$
R_o	$R_1 + \ldots + R_C$

Table 3-1: Formulae for Open Queueing Networks with UAP.

chain with its workload contributed from both the main chain and the UAP chain; section 3.3 discusses the formulation of useful performance measures for open queueing networks with UAP. On the other hand, if the original network is a closed network, then we have a mixed network with the closed main chain and the open UAP chain. Section 3.4 discusses the necessary and sufficient condition for the closed network to be stable and an iterative procedure which computes the system throughput.

It is extricable now to formulate networks with UAP. Let the summation of visit ratios over all the cuts in section 3.2.A, V(U), denote the "internally generated" visit rate of the UAP chain. Note that "(M)" will denote an open chain in section 3.3 and a closed chain in section 3.4.

3.3 OPEN QUEUEING NETWORKS WITH UAP

For an open queueing network with UAP, the network arrival process is assumed to be poisson with a constant rate λ_o. By solving the extended routing matrix introduced in section 2, one can obtain the visit ratios for all classes, hence V(U). Since λ_o is given, $X_o(U)$ is also determined, specifically, $X_o(M) = \lambda_o$ and $X_o(U) = \lambda_o * V(U)$. For instance suppose $\lambda_o = 5$ customers/sec in Figure 1-2, then the UAP chain (SOURCEU, WD1, WD2, SINKU) ,as shown in Figure 1-4, has an arrival rate of 1 customer/sec.

Since the network can be aggregated to an open single chain network, its stability follows from OPFSCQN, i.e. the network is stable if and only if $U_i < 1$ for all facilities in the network.

balanced main chain with its classes in the set $\{R_c\}$ and many open chains with their classes in the set $\{R\} - \{R_c\}$. Therefore, the classes in the main chain and the classes in the open chains are disjoint.

However, it has been pointed out, in section 2, that a split may split again, so the open chains may themselves be flow unbalanced. To solve the problem, it is logical to cut all the additional unbalanced flows in the open chains continuously (and insert "internal sources" which generate equivalent flow rates as those of the open chains before the cuts) until all flows are balanced, forming additional open chains.

It is assumed that service time distributions and service disciplines of the facilities in the network follow those of classical product form queueing networks (Baskett et al); in addition, the unbalanced flows which run independently of one another are assumed to arrive at their destinations as independent poisson processes (this assumption is also adopted by Goldberg (1983) and Heidelberger (1983). However, as Burke (1972) pointed out, these processes are not Poisson in general. The simulation studies that the authors have conducted indicate that this is a fairly robust approximation. The validation reported by Goldberg, et al (1983) provides further support for this assumption. It follows that the OPFMCQN result can be applied to aggregate the additional open chains discussed in the last paragraph to a single open chain -- the UAP chain.

If the original network is an open network, then the OPFMCQN result can be applied again to make the overall network a single

main chain; and $D_i(M) = S_i(M) * V_i(M)$ is the product of visit ratio and mean service time of facility i for the main chain.

3.2 QUEUEING NETWORKS WITH UAP

It was noted, in section 2, that a) UAP can occur in many classes within a queueing network; b) an input to a class that causes UAP may be the output from another UAP class; and c) all the additional unbalanced flows are defined to belong to the UAP chain -- a single chain. It is natural to ask whether the flows of the transformed network would be balanced, and what kind of relationship would exist between the main chain and the UAP chain. These questions are answered below:

If one cuts the additional b-1 unbalanced flows from a class which is UAP with degree b and inserts "internal sources" (SOURCEU) which generate customers with equivalent flow rates as those of the network before the cut, then following the assumption that unbalanced flows run independently of one another except for resource contention, the b-1 unbalanced flows will form b-1 new open chains which will not interact with the main chain. If all the additional unbalanced flows (spawned from the classes which are UAP and connected to the main chain) are cut from the main chain, then the flow in the main chain will be balanced, as illustrated in Figure 1-3.

Let {R} denote the set of classes in the network before the cuts and $\{R_c\}$ denote the set of classes in the main chain, as illustrated in Figure 2-1 and 2-2. It follows that we have the

C	total number of classes in the network.
CMD	continuous and monotonically decreasing
D	V*S; the product of visit ratio and mean service time.
FCFS	first come first serve.
f	$X_0(M)=f(X_0(U))$; the main chain throughput as a nonlinear function of the UAP chain throughput.
IS	infinite server.
LCFSPR	last come first serve preemptive resumable.
M	number of service facilities in the network.
N	mean number of customers(mean queue length including the one in service).
n	number of customers.
PS	processor sharing.
p.f.s.	product form solution.
p.g.f.	probability generating function.
R	mean response time.
S	mean service time.
U	utilization.
UAP	unbalanced flows due to asynchronously spawned parallel tasks.
V	visit ratio.
X	throughput.
λ	arrival rate.
ρ	traffic intensity.

Example: $S_i(M)$ means the mean service time of facility i for the main chain; $V_i(M)$ means the visit ratio to facility i due to the

3. ANALYTIC FORMULATION

To present the paper concisely, service rate is assumed to be fixed, i.e. load independent (Denning and Buzen, 1975, Lavenberg, 1983 or Lazowska, 1984). The following useful results and models (Kleinrock, 1975, 1976, or Lavenberg, 1983) are recapitulated in the Appendix to serve as a point of departure for the discussion in this section: 1) Little's formula, 2) non-product-form queueing networks, 3) product form queueing networks, 4) single chain queueing networks, 5) Open product form single chain queueing networks (OPFSCQN), 6) Open product form multiple chain queueing networks (OPFMCQN), 7) closed product form single chain queueing networks (CPFSCQN), 8) Convolution algorithm, and 9) product form mixed queueing networks (PFMQN).

3.1 NOTATIONS

Notations used in this paper are listed below:

A) subscripts:

i denotes an individual service facility.

o denotes the overall network.

(M) denotes the main chain.

(U) denotes the UAP chain.

$()^i$ denotes the ith iteration.

B) notations:

B bottleneck facility (therefore chain) throughput.

$$R = \begin{array}{l} \\ \text{SOURCEM} \\ \text{RP1} \\ \text{RD1} \\ \text{WP1} \\ \text{WD1} \\ \text{WD2} \end{array} \begin{array}{c} \begin{array}{ccccccc} \text{RP1} & \text{WP1} & \text{RD1} & \text{SINKM} & \text{WD1} & \text{WD2} & \text{SINKU} \end{array} \\ \begin{bmatrix} .8 & .2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Figure 2-1: The Extended Routing Matrix for Figure 1-2

$$R_c = \begin{array}{l} \\ \text{SOURCEM} \\ \text{RP1} \\ \text{RD1} \\ \text{WP1} \end{array} \begin{array}{c} \begin{array}{cccc} \text{RP1} & \text{WP1} & \text{RD1} & \text{SINKM} \end{array} \\ \begin{bmatrix} .8 & .2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Figure 2-2: The Unextended Routing Matrix for Figure 1-2

The visit ratios of the classes in R_c can be obtained from the visit ratio equations (5, p.237), viz.,

$$V_j = p_{o,j} + \sum_{i=1}^{C} V_i * p_{i,j} \qquad j = 1, \ldots, C.$$

Alternatively, the visit ratio equations can be applied directly to the extended routing matrix R to obtain all the visit ratios of the classes in R.

with a part of the main chain. Therefore, a unique syntactic definition exists for each UAP network.

Classical queueing network models cannot be applied to analyze UAP directly because of the unbalanced flows mentioned. An extended routing matrix is introduced below to accommodate the problem. Let R denote the extended routing matrix of an UAP network where a row-sum may be greater than one. The extended routing matrix R for Figure 1-2 is shown in Figure 2-1. Let R_c denote the unextended routing matrix which excludes the UAP chain of the network. The unextended routing matrix R_c which excludes the UAP chain (SOURCEU, WD1, WD2, SINKU) is shown in Figure 2-2. Elements in R and R_c are the routing probabilities $p_{i,j}$'s.

Define the visit ratio of a class, V_c, as the mean number of requests of the class to a service facility per customer. Define the sum of visit ratios of all exogenous sources, V_o, in an open system to be one. In a closed system, the outputs feedback to the system inputs; the sum of visit ratios of the system inputs is also defined to be one.

For example, in Figure 1-2, UAP(WP1,P1) = 2. Note that (a) UAP can occur in many classes within a queueing network, for instance acknowledgements may take place at different levels of a data storage hierarchy; and (b) the inputs to a class that cause UAP can be the outputs from other UAP classes. For instance, a split from an acknowledgement may split again to send more acknowledgements to other classes.

Consider a class which is UAP with degree b, the main task that eventually returns to the reference source is defined to belong to the main chain; on the other hand, the b-1 additional flows which cause that class to be unbalanced are perceived as "internal sources" (denoted as SOURCEU) which generate customers to travel within the network and eventually terminate at the "internal sink" (denoted as SINKU). It follows, as justified in section 3, that all the classes with UAP can be separated from the main chain to form the UAP chain where the <u>UAP chain</u> is defined as the additional path through which the "internally generated" customers (from SOURCEU) travel and eventually sink (at SINKU). For example, in Figure 1-4, the classes (SOURCEU, WD1, WD2, SINKU) define the UAP chain. Note that SOURCEU may stand for multiple "internal sources".

By labeling the source and sink of the main chain as SOURCEM and SINKM, and the source and sink of the UAP chain as SOURCEU and SINKU, one can decompose the graph of a network model with UAP unambiguously without referring to the semantics of the model. In other words, given the labeled graph of an UAP network, it is impossible to interchange one of the UAP flows

2. MODEL STRUCTURE

Without loss of generality, let's assume that all customers are homogenous, i.e. there is a single customer type. In Figure 1-2, the single type customer has a 0.8 probability of requesting the read operation and a 0.2 probability of requesting the write operation. It would be easy to relax this assumption to include different types of customers.

Let there be M service facilities and C classes in a queueing network. A service facility may consist of several classes which allow customers to have different sets of routing probabilities for different visits. Assume that any sources and sinks belong to class 0. Let $p_{i,j}$ denote the routing probability which is the fraction of the customers completing service in class i that joins class j. i = 0, ..., C; j = 0, ..., C; and $p_{0,0} = 0$ by convention.

A <u>main chain</u> is defined as the path through which customers travel according to the defined routing probability and eventually go out of the system to return to the reference source. Since all customers have been assumed to be homogeneous, there is only one main chain in the system. In Figure 1-3, the classes (SOURCEM, RP1, RD1, WP1, SINKM) define the main chain.

A <u>class c customer of facility m</u> in the queueing network is said to be <u>UAP with degree b</u> ,i.e. UAP(c,m)=b, if its output splits into b branches where b is a real number greater than one but each branch has a routing probability not greater than one.

approximate solution method was developed and results of the approximation were compared to those of simulations. Mean value analysis approximation techniques were proposed for local area distributed computer systems with UAP by Goldberg, Popek, and Lavenberg (1983).

It is perhaps interesting to note at this point that, quite independently from the above research, the authors of this paper developed what they have called "Distributed Systems with Unbalanced Flows" (Wang and Madnick, 1981, 1984) starting in 1981. The technique used to model UAP is very similar but a different algorithm is used to test the necessary and sufficient condition as well as to compute the closed network throughput. Moreover, the results for open networks with UAP, such as response time, have been analyzed. A syntactic definition has also been given to decompose a model uniquely.

A terminal-oriented system and a batch-oriented multiprogramming system are modeled by Heidelberger (1982), and local area distributed systems are modeled by Goldberg and others (1983) while a hierarchically decomposed architecture is modeled in the INFOPLEX research (Wang and Madnick, 1984). The consistency reported from modeling these different architectures provides further validation of the modeling technique. The UAP model is described in the next section.

property, the convolution algorithm can be applied to solve the nonlinear equation iteratively. Let $()^i$ denote the ith iteration. For instance, $(EN(X_o))^5$ denotes the throughput of EN at the 5th iteration. In the iterative procedure, $(X_o(U))^0$ is estimated initially by the lemma shown later and $(X_o(U))^{i+1}$ is determined as follows: $(X_o(U))^{i+1} = (EN(X_o))^{i+1} * V(U)$ where $(EN(X_o))^{i+1} = f((X_o(U))^i)$. This relationship is used below to settle issue B.

B) The stability of PFMQN is unaffected by the presence of closed chains (Lazowska, 1984). Define the facility with maximum open chain utilization (i.e. the bottleneck facility) as facility I. It follows that a closed network with UAP is stable if and only if $U_i(U)<1$. Note that $U_I(U) = (X_o(U) / V(U)) * D_I(U)$. Denote $V(U)/D_I(U)$ as B, the maximum throughput of the bottleneck facility, i.e., at saturation. It follows that a closed queueing network with UAP is stable if and only if $X_o(U) < B$.

Denote $D_I(M)$ as the main chain D value at the bottleneck facility I.The stability condition of the closed network with UAP can then be identified with the following four mutually exclusive and collectively exhaustive cases:

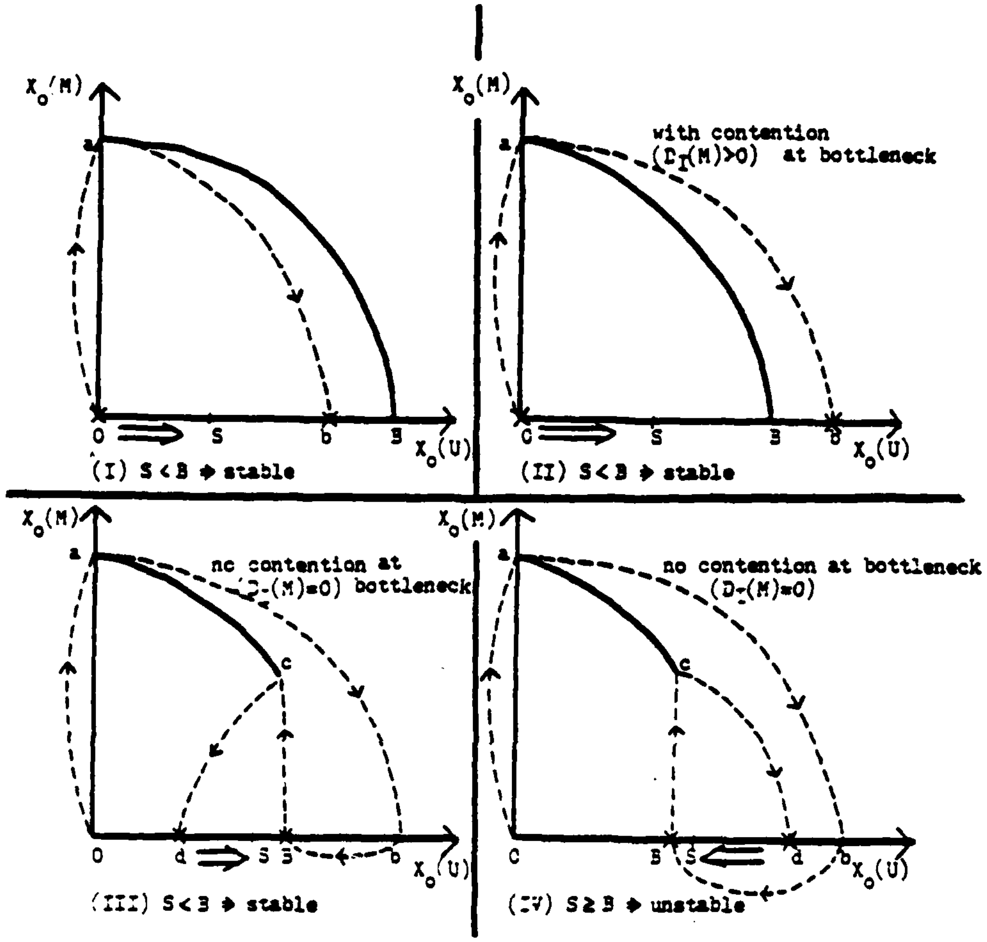
I) $f(X_o(U)=0) * V(U) < B$;

II) $f(X_o(U)=0) * V(U) \geq B$, but $D_I(M) > 0$;

III) $f(X_o(U)=0) * V(U) \geq B$, $D_I(M) = 0$, but $f(X_o(U)=B) * V(U) < B$;

IV) $f(X_o(U)=0) * V(U) \geq B$, $D_I(M) = 0$, and $f(X_o(U)=B) * V(U) \geq B$.

Figure 3-3 depicts the four conditions and the lemma below establishes the condition for stability.



<u>Key</u>

a = $f(X_o(U)=0)$ b = $f(X_o(U)=0)*V(U)$

c = $f(X_o(U)=B)$ d = $f(X_o(U)=B)*V(U)$

B: bottleneck throughput

S: solution

$X_o(M)$: Main chain throughput   ×—S—× : bound

$X_o(U)$: UAP chain throughput

<u>Figure 3.3</u> <u>Four Cases to Test the Stability Condition</u>

Let $a = f(X_o(U)=0)$, $b = a*V(U)$, $c = f(X_o(U)=B)$, and $d = c*V(U)$, then the four cases can be rewritten as follows:

I) $b < B$; II) $b \geq B$, but $D_I(M) > 0$;

III) $b \geq B$, $D_I(M) = 0$, but $d < B$;

IV) $b \geq B$, $D_I(M) = 0$, and $d \geq B$.

Lemma: The network is stable if and only if it is not case IV.

Proof: Case I states that zero is given as the initial estimate for $(X_o(U))^0$, and $(X_o(U))^1 = (EN(X_o))^1 * V(U) = b < B$, as shown in Figure 3-3.I. Since f is CMD and a is the upper bound of the main chain throughput, it follows that $(X_o(U))^1$ is bounded between 0 and b for all i. Therefore, the stability condition holds since $b < B$.

Case II states that zero is given as the initial estimate for $(X_o(U))^0$, but $(X_o(U))^1 \geq B$ as shown in Figure 3-3.II. Since there exists contention at the bottleneck facility I, i.e. $D_I(M) > 0$, the actual throughput, $(X_o(U))^\infty$, will be less than $(X_o(U))^1$. Suppose a solution exists between B and b, i.e. $B \leq (X_o(U))^\infty = (EN(X_o))^\infty * V(U) \leq b$. It follows that $(EN(X_o))^\infty \geq B/V(U) > 0$. On the other hand, there exists contention at facility I, therefore $(EN(X_o))^\infty = 0$ because the bottleneck facility I is fully utilized by the open UAP chain, blocking the closed main chain flow completely. However, this is contradictory to the supposition; therefore, the solution is bounded in the open interval (0,B) which is less than B and the stability condition holds.

Case III states that there is no contention at the bottleneck facility, and if B is given as the initial estimate for $(X_0(U))^0$, then $(X_0(U))^1 = d < B$ as shown in Figure 3-3.III. It follows, by CMD, that a solution exists in the open interval (d,B); therefore the stability condition holds.

Case IV states that $D_I(M) = 0$ which implies that the bottleneck facility I does not contribute to the main chain throughput at all. The only impact it might have is to cause the overall network to be unstable. If B is given as the initial estimate for $(X_0(U))^0$, $(X_0(U))^1 = d \geq B$. It follows, by CMD, that if a solution exists, it must be greater than or equal to B. Thus, case IV violates the stability condition. Q.E.D.

Several points are worth noting:

a) The IS discipline was excluded since the main chain and the UAP chain do not interact with each other at the IS facility. For networks with mixed disciplines, the inflating factor for the IS facility is one. For networks with IS facilities only, the UAP chain has no impact on the main chain, therefore, can be ignored.

b) The Convolution algorithm, simple and efficient, is used to insure the stability condition as well as to locate the solution (Wang and Madnick, 1984).

c) The equivalent closed network obtained from the corollary is used to calculate the "system response time" perceived by the external world. Moreover, when the iterative procedure stops, G(1), ..., G(N) are also available as a byproduct for calculating useful performance measures.

d) It was found (Wang and Madnick, 1984) that a bounded interpolation algorithm takes an average of 2.4 iterations to produce relative errors less than 0.001 over 10,000 simulations.

e) A comparative study of the INFOPLEX data storage hierarchy has been conducted to assess the predictability of this technique. It has been observed (Wang and Madnick, 1984) that the analytic results are highly consistent with the simulations. A closer examination of the data shows that the analytic results deviate from the simulations by less than 2%.

4. CONCLUSIONS

An analytic approximation methodology has been developed to model distributed information systems with unbalanced flows due to asynchronously spawned tasks (UAP). The methodology allows us to assess useful performance measures which are crucial to the success of information systems utilizing distributed processing or local area networking. It equips system designers with a cost effective tool to explore different design alternatives. Whereas, it may not be possible to attain the steady-state results of a single design alternative using simulation economically. Studies have shown (Wang and Madnick, 1984) that this methodology produces the same quality of results as simulation with less effort in addition to the cost effectiveness.

Several areas are open for researchers to extend the work: 1) UAP networks with priorities; 2) software deadlocks; and 3) hierarchical decomposition of distribution systems with UAP. The payoff will be significant for multi-million dollars are usually at stake in a distributed information system project. Extending the work to the above areas will provide information system designers with an even more powerful set of tools to determine system performance early in the development process.

5. APPENDIX

1 Little's Formula (22)

Let N be the average over all time of the number of customers in a system, λ be the average arrival rate at the system, and R be the average over all arrivals at the system of the system response time, then $N = \lambda * R$. This formula states that the average number of customers in the system is equal to the product of the arrival rate and the average system response time.

2 Non-Product-Form Queueing Networks

If a queueing network model does not have a product form solution, then we usually must use fairly general numerical techniques, such as solution of Markov balance equations, for its solution. In this case we shall find the exact solution of the network intractable unless it has few service facilities and/or customers (20).

3 Product Form Queueing Networks

For the following queueing disciplines, a product form solution exists for a queueing network: first come first serve (FCFS), processor sharing (PS), infinite server (IS), and last

come first serve preemptive resumable (LCFSPR). If a server has a PS, IS, or LCFSPR discipline, then different service time distributions are allowed for different classes at a service facility. In this cases, the service time distributions affect the performance measures we shall consider only through the mean service time. If a service facility has a FCFS discipline, then all classes at the facility must have the same exponential service time distribution (1, 16, 17, 20). A product form queueing network is one that has a solution of the following form:

$$P(S_1,\ldots,S_M) = P_1(S_1) \ldots P_M(S_M)/G(N)$$

where $P(S_1, \ldots S_m)$ is the steady-state probability of a network state in a network with M service facilities, $P_m(S_m)$, m = 1, ..., M is a factor corresponding to the steady-state probability of the state of service facility m in isolation. N is the number of customers in the network, and G(N) is a normalization constant. For an open system, N can be any number; for a closed system, N is a fixed number of customers in the system. The normalization constant G(N) is equal to the sum of $P_1(S_1) * \ldots * P_m(S_m)$ over all feasible network states.

4 Single Chain Queueing Networks

A single chain queueing network is one with only one

customer type. However, service facilities may have several classes which allow customers to have different sets of routing probabilities for different visits to a service facility. Note that although there are several classes and several routing probabilities, the only parameters in the product form solutions, when aggregated to the service facility level, are visit ratios, mean service times, and number of customers in the closed queueing network case (20).

5 Open Product form Single Chain Queueing Networks(OPFSCQN)

An OPFSCQN is one with M service facilities and C classes and a single chain that has a product form solution. In addition, there are sources for exogenous arriving customers and sinks for departing customers. It is assumed that customers from exogenous sources form a poisson process with a constant arrival rate λ.

A remarkable theorem by Jackson states that for OPFSCQN with a constant arrival rate, the network is separable (15), i.e. one can compute a service facility's performance measures as follows (20): Suppose the probability that an arrival customer enters class c is $P_{o,c}$ then it must be true that

$$\sum_{i=1}^{C} P_{o,j} = 1$$

$$V_j = p_{o,j} + \sum_{i=1}^{C} V_i * p_{i,j} \qquad j = 1,\ldots,C$$

Suppose the system is in the steady-state, then the system arrival rate is equal to departure rate. Let X_0 denote system throughput, it follows that $X_0 = \lambda$. Let X_i be the throughput of facility i, it follows that $X_i = X_0 * V_i \quad i = 1, \ldots, M$. Let $U_i = X_i * S_i$ where U_i is the utilization of service facility i and S_i is the mean service time of facility i. It is easy to see that an open queueing network is stable iff $U_i < 1$ for all service facilities in the network. The IS discipline is excluded from our discussion to avoid unnecessary digression. The mean queue length(including the one in service) is $N_i = U_i / (1-U_i)$.

By Little's formula, the mean response time of service facility i is $R_i = N_i / X_i$. It follows that system response time $R = R_1 + \ldots + R_M$. The mean number of customers in the network $N = R / X_0$. Note that different formulae should be used for the IS discipline. Thus, for OPFSCQN, one can obtain system as well as facility throughput, response time, and mean queue length.

6 Open Product Form Multiple Chain Queueing Networks (OPFMCQN)

OPFSCQN have a single source and a single sink and all classes are reachable from the source and the sink is reachable from all classes. It is not necessary, however, that all classes be reachable from one another. If there are H sources and the classes are partitioned into H disjoint subsets such that for h = 1, ..., H, all classes in subset h are reachable from source h and not reachable from any other sources or any other classes in

any other subsets, then there are H open routing chains (20). It can be shown that (20, 31, 33) that if we have H chains, each with a poisson source with constant rate λ_h , h = 1, ..., H, then we can treat the H open chains as a single aggregate chain if we give that aggreate chain an arrival rate $\lambda = \lambda_1 + \ldots + \lambda_H$, and where class c belongs to chain h in the original network, make the replacement $P_{0.c} = (\lambda_h/\lambda) * P_{0.c}$, c = 1, ...,C.

7 Closed Product Form Single Chain Queueing Networks (CPFSCQN)

A closed product form single chain queueing network is one with M service facilities, C classes, and a fixed number of homogenous customers that has a product form solution. Several algorithms are available for CPFSCQN, the convolution algorithm (4) remains the dominant algorithm for general purpose use (20).

The equilibrium distribution of customers in CPFSCQN, aggregated at the service facility level, is given by:

$$P(n_1, \ldots, n_m) = (1/G(N)) * \prod_{i=1}^{M} (D_i)^{n_i}$$

where $D_i = V_i \cdot S_i$, and n_i is the number of customers of facility i. It can be shown(5) that

$$P(n_i = k) = (D_i)^k (G(N-k) - D_i * G(N-k-1))/G(N)$$

where G(n) is defined as zero for n<0.

The mean queue length of facility i, N_i, is given by

$$N_i = \sum_{k=1}^{N} (D_i)^k * G(N-k) \;/\; G(N)$$

The system throughput, X_o, is given by $X_o = G(N-1)/G(N)$. Therefore, once the values of G(1), ..., G(N) are given, a number of useful performance measures can be computed.

8 Convolution Algorithm

The expression for G(N) in the equilibrium distribution equation involves the summation of C(M+N-1,N) terms, each of which is a product of M factors which are themselves powers of the basic quantities. However, the celebrated Convolution algorithm computes the entire set of values G(1), ..., G(N) using a total of N*M multiplications and N*M additions. The implementation of the algorithm is extremely simple:

```
/* Initialization */
G(0) = 1
for n = 1 to N
G(n) = 0
/* convolution */
for m = 1 to M
for n = 1 to N
G(n) = G(n) + D(m)*G(n-1)
/* end convolution */
```

9 Product Form Mixed Queueing Networks (PFMQN)

Let's restrict a product form mixed queueing network to be one with only one closed chain and one open chain. Let "(C)" denote the closed chain, and "(O)" denote the open chain. The traffic intensities of facility i due to the open chain and the closed chain are defined as

$$\rho_i(O) = X_0(O) * V_i(O) * S_i(O)$$

$$\rho_i(C) = X_0(C) * V_i(C) * S_i(C)$$

The p.g.f. method has been used by Reiser and Kobayashi (31) to provide important theoretical results for PFMQN. It is found, with the p.g.f. method, that 1) The stability of PFMQN is unaffected by the presence of closed chains; 2) The open and the closed chains do not interact at an IS service facility; and 3) The closed chain throughput is evaluated through a nonlinear function of the open chain throughput.

6. REFERENCES

1. Baskett, F., Chandy, K. M., Muntz, R. R., and Palacios, J. "Open, Closed, and Mixed Networks with Different Classes of Customers," JACM, April 1975, pp. 248-260.

2. Browne, J. C., Chandy, K. M., Hogarth, J., and Lee, C. "The Effect on Throughput in Multi-processing in a Multi-programming environment," IEEE Trans. Computers, Vol. 22, August 1973, pp. 728-735.

3. Burke, P. J. "Output Process and Tandem Queues," Symposium on Computer-Communications Networks and Teletraffic, Polytechnique Institute of Brooklyn, PP. 419-428, 1972.

4. Buzen, J. P. "Computational Algorithms for Closed Queueing Networks with Exponential Servers," CACM, Sept. 1973, pp. 527-531. Cambridge, MA, 1971.

5. Denning P. J. and Buzen, J. P. "The Operational Analysis of Queueing Network Models," ACM Computing Surveys, Vol.10, No. 3, Sept. 1978, pp. 225-261.

6. Gagliardi, Ugo, Lectures on Software Engineering, Harvard University, 1982.

7. Geist, R. M., and Trivedi, K. S. "Optimal Design of Multilevel Storage Hierarchies," IEEE Transactions on Computers, March 1982.

8. Goldberg, A. ,Popek, G., and Lavenberg, S. "A Validated Distributed System Performance Model," Performance'83, pp. 251-268.

9. Gordon, W. J. and Newell, G. F. "Closed Queueing Systems with Exponential Servers," Operation Research 15 (1967), pp. 254-265.

10. Goyal, A. and Agerwala, T. "Performance Analysis of Future Shared Storage Systems," IBM J. Res. Develop., January 1984.

11. Graham, G. S. "Guest Editor's Overview: Queueing Network Models of Computer System Performance," ACM Computing Surveys, Vol. 10, #3, September 1978, pp 219-224.

12. Hamming, R. W. "Numerical Methods for Scientists and Engineers," 2nd ed. New York: Mc Graw-Hill, 1973.

13. Heidelberger, P. and Trivedi, K. S. "Analytic Queueing Models for Parallel Processing with Asynchronous Tasks," IEEE Transactions on Computers, November 1982.

14. Heidelberger, P. and Trivedi, K. S. "Analytic Queueing Models for Programs with Internal Concurrency," IEEE Transactions on Computers, January 1983.

15. Jackson, J. R. "Jobshop Like Queueing Systems," Management Science 10 (1963), pp. 131-142.

16. Kleinrock, L. "Queueing Systems I," John Wiley, New York, 1975

17. Kleinrock, L. "Queueing Systems II," John Wiley, New York, 1976

18. Lam, C. Y., "Data Storage Hierarchy Systems for Database Computers," Tech. Rep. #4, August 1979, MIT Sloan School.

19. Lam, C. Y. and Madnick, S. E., "Properties of Storage Hierarchy Systems with Multiple Page Sizes and Redundant Data," ACM Transactions on Database Systems, Vol. 4, No. 3, September 1979, pp. 345-367.

20. Lavenberg, S. S. "Computer Performance Modeling Handbook," Academic Press, 1983.

21. Lazowska, Zahorjan, Graham, and Sevcik, "Quantitative System Performance: Computer System Analysis Using Queueing Network Models," Prentice Hall, 1984.

22. Little, J. D. C. "A Proof of the Queueing Formula $L=\lambda W$," Operations Research 9, 383-387 (1961)

23. Madnick, S. E., "Storage Hierarchy Systems," Report No. TR-105, Project MAC, MIT, Cambridge, MA, 1973.

24. Madnick, S. E., "Trends in Computers and Computing: The Information Utility," Science, Vol. 185, March 1977, pp. 1191-1199.

25. Madnick, S. E., "The INFOPLEX Database Computer, Concepts and Directions," Proc. IEEE Comp. Con., February 1979, pp. 168-176.

26. Maekawa, M and Boyd, D. L. "Two Models of Task Overlap with Jobs of Multiprocessing Multiprogramming Systems," Proc. 1976 Int. Conf. on Parallel Processing, Detroit, August 1976, pp. 83-91.

27. Muntz, R. R. "Poisson Departure Processes and Queueing Networks," IBM Res. Rep. RC-4145, 1972.

28. Mayrhauser, A. V. and Kenevan, J. R. "Convexity of Reciprocal Throughput and Response Time for a Very General Class of Queueing Networks," Illinois Institute of Technology, June 1983.

29. Peterson, M. and Bulgren, W. "Studies in Markov Models of Computer Systems," Proc. 1975 ACM Annual Conf., Minneapolis, Minn., pp. 102-107.

30. Price, T. G. "Models of Multiprogrammed Computer Systems with I/O buffering," Proc. 4th Texas Conf. Comput. Syst., Austin, 1975.

31. Reiser, M. and Kobayashi, H. "Queueing Networks with Multiple Closed Chains: Theory and Computational Algorithms," IBM J. Res. Develop., May 1975, pp. 283-294.

32. Reiser, M. and Chandy, K. M. "The Impact of Distributions and Disciplines on Multiple Processor Systems," CACM, Vol.22, pp. 25-34, 1979.

33. Sauer, C. H. and Chandy, K. M. "Computer Systems Performance Modeling," Printice-Hall, Englewood Cliffs, New Jersey, 1981.

34. Sauer, C. H., MacNair, E. A., and Kurose, J. F. "The Research Queueing Package Version 2: Introduction and Examples," RA 138, 4/12/82 IBM Thomas J. Watson Research Center, Yorktown Heights, New York.

35. Sevcik, K. C. and Mitrani, "The Distribution of Queueing Network States at Input and Output Instants,"

36. Towsley, D., Chandy, K. M., and Browne, J. C. "Models for Parallel Processing within Programs: Applications to CPU:I/O and I/O:I/O Overlap," CACM, Vol.21, pp. 821-831, 1978.

37. Trivedi, K. S. and Sigmon, T. M. "Optimal Design of Linear Storage Hierarchies," JACM April 1981, pp. 270-288.

38. Wang, Y. R. and Madnick, S. E. "Performance Evaluation of the INFOPLEX Database Computer," Tech. Rep. #13, MIT Sloan School, April 1981.

39. Wang, Y. R. and Madnick, S. E. "Performance Evaluation of Distributed Systems with Unbalanced Flows: An analysis of the INFOPLEX Data Storage Hierarchy," Tech. Rep. #15, MIT Sloan School, July 1984.

END

FILMED

9-85

DTIC